CLAIMS:

1. A method for recovering a global history vector in the event of a non-branch flush comprising the steps of:

receiving a tag, wherein said tag is associated with a flush operation;

resetting a next-to-write pointer in a queue using said received tag;

reading a global history vector from said queue in an entry indexed by said tag;

inserting a copy of said global history vector read into a global history vector mechanism, wherein said global history vector mechanism is configured to update said global history vector in a first mode; and

transmitting a command to said global history vector mechanism to enter a second mode if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer, wherein said global history vector mechanism does not update said global history vector in said second mode.

2. The method as recited in claim 1, wherein said command is transmitted to said global history vector mechanism to enter said second mode if said tag points to an entry in said queue containing information regarding a branch instruction previously fetched between an address flushed to and a last address fetched at a time of said flush operation.

3. The method as recited in claim 1, wherein if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer then the method further comprises the steps of:

scanning a copy of a selected instruction; and

determining if said selected instruction is a branch instruction;

wherein if said first instruction is not said branch instruction, then the method further comprises the step of:

scanning a copy of a next instruction.

4. The method as recited in claim 3, wherein if said selected instruction is a branch instruction, then the method further comprises the step of:

transmitting a command to said global history vector mechanism to enter said first mode of operation.

5. The method as recited in claim 4, wherein if said first instruction is said branch instruction, then the method further comprises the step of:

updating said global history vector upon each successive fetch of a group of instructions in said first mode.

6. The method as recited in claim 1, wherein if said flush operation is a flush to a group of instructions that contains a branch instruction then the method further comprises the step of:

transmitting a command to said global history vector mechanism to continue operation in said first mode.

7. The method as recited in claim 1, wherein if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer then the method further comprises the steps of:

setting a counter to a predetermined value;

transmitting an instruction; and

decrementing said counter if said instruction transmitted is not a branch instruction.

8. The method as recited in claim 7, wherein if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer then the method further comprises the step of:

determining if said counter equals zero, wherein if said counter equals zero then the method further comprises the step of:

transmitting a command to said global vector history mechanism to reconstruct said global history vector.

9. The method as recited in claim 8, wherein if said counter equals zero then the method further comprises the step of:

reconstructing said global history vector by appending a number of zero's in said global history vector, wherein said number equals said predetermined value.

10. The method as recited in claim 9, wherein if said counter equals zero then the method further comprises the step of:

continuing reconstructing said global history vector by updating said global history vector upon each successive fetch of a group of instructions.

11. A system, comprising:

an instruction fetch unit configured to fetch instructions, wherein said instruction fetch unit comprises:

a branch prediction logic unit, wherein said branch prediction logic unit is configured to predict if a branch instruction will be taken or not taken, wherein said branch prediction logic unit comprises:

a group history vector mechanism configured to manage a group history vector; and

a queue coupled to said branch prediction logic unit, wherein said queue is configured to store information about branch instructions, wherein said queue comprises:

logic for receiving a tag, wherein said tag is associated with a flush operation;

logic for resetting a next-to-write pointer in said queue using said received tag;

logic for reading said global history vector from said queue in an entry indexed by said tag; and

logic for transmitting a command to said global history vector mechanism to enter a second mode if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer, wherein said global history vector mechanism does not update said global history vector in said second mode.

12. The system as recited in claim 11, wherein said command is transmitted to said global history vector mechanism to enter said second mode if said tag points to an entry in said queue containing information regarding a branch instruction previously fetched between an address flushed to and a last address fetched at a time of said flush operation.

13. The system as recited in claim 11, wherein said instruction fetch unit further comprises:

a branch scan logic unit coupled to said branch prediction logic unit, wherein said branch scan logic unit comprises:

logic for scanning a copy of a selected instruction if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer;

logic for determining if said selected instruction is a branch instruction; and

logic for scanning a next instruction if said first instruction is not said branch instruction.

14. The system as recited in claim 13, wherein said branch scan logic unit further comprises:

logic for transmitting a command to said global history vector mechanism to enter said first mode if said first instruction is said branch instruction.

15. The system as recited in claim 14, wherein said global history vector mechanism comprises:

logic for updating said global history vector upon each successive fetch of a group of instructions in said first mode.

16. The system as recited in claim 11, wherein said queue further comprises:

logic for transmitting a command to said global history vector mechanism to continue operation in said first mode if said flush operation is a flush to a group of instructions that contains a branch instruction.

17. The system as recited in claim 11, wherein said global history vector mechanism comprises:

logic for setting a counter to a predetermined value if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer;

wherein said instruction fetch unit further comprises:

a cache memory, wherein said cache memory comprises:

logic for transmitting an instruction if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer; and

wherein said global history vector mechanism further comprises:

logic for decrementing said counter if said instruction transmitted is not a branch instruction.

18. The system as recited in claim 17, wherein said instruction fetch unit further comprises:

a branch scan logic unit coupled to said cache memory, wherein said branch scan logic unit comprises:

logic for determining if said counter equals zero if said flush operation is a flush to a group of instructions that contains no branch instructions and said tag does not equal said next-to-write pointer; and

logic for transmitting a command to said global vector history mechanism to reconstruct said global history vector if said counter equals zero.

19. The system as recited in claim 18, wherein said group history vector mechanism further comprises:

logic for reconstructing said global history vector by appending a number of zero's in said global history vector if said counter equals zero, wherein said number equals said predetermined value.

20. The system as recited in claim 19, wherein said group history vector mechanism further comprises:

logic for continuing reconstructing said global history vector by updating said global history vector upon each successive fetch of a group of instructions if said counter equals zero.